# Porting NuttX to STM32F401RC-RS485

# What you need to do the port?

- Schematics: (why?)
  - Clock used: crystal freq., no crystal at all (internal oscillator), etc
  - External peripherals connected to MCU
  - Serial pins ( U(S)ART TX/RX pins)
  - Programming pins (SWD / JTAG)
- https://github.com/lucaszampar/NuttX_STM32F4_RS485_DevBoard/blob/main/PDF/NuttX_STM32F4_RS485.pdf
- Microcontroller datasheet
- Datasheets of chips used in the board

# First Steps to Board Port (Requirements)

- NuttX source code
  - Kernel: https://github.com/apache/nuttx
  - Apps: https://github.com/apache/nuttx-apps

- GCC or CLANG Toolchain for you MCU (microcontroller)

# Find a board with similar microcontroller

- Clear the environment:
  - cd nuttx/
  - make distclean
- Use a similar board as starting point (with same MCU or similar)
  - cd boards/<arch of your board>/<chip family of your board>/
    - i.e.: cd boards/arm/stm32/
  - git grep MCU_OF_YOUR_BOARD
    - i.e.: git grep STM32F401

# Start copying from a similar board

- Use the board with similar MCU as start point (copy it to you)
  - i.e.: $ cp -a nucleo-f4x1re stm32f401rc-rs485
- Enter inside this new create directory and confirm it contains:
  - $ ls

```
CmakeLists.txt configs include Kconfig scripts src
```

# Create or copy a board.h

- Enter inside stm32f401rc-rs485/include and remove the files:
  - $ ls

    ```
    board.h nucleo-f401re.h nucleo-f411re.h
    ```
  - $ rm *
- Now copy the board.h from stm32f411e-disco because it uses the same 8MHz clock as our STM32F401 board:
  - $ cp ../../stm32f411e-disco/include/board.h .

# Fix your Microcontroller Flash/RAM Size

- Enter inside stm32f401rc-rs485/scripts and fix the Flash/RAM:
  - $ cd ../scripts/
  - Open ld.script to setup fix Flash and RAM size for STM32F401RC:

```
MEMORY
{
    flash (rx) : ORIGIN = 0x08000000, LENGTH = 128K
    sram (rwx) : ORIGIN = 0x20000000, LENGTH = 64K
}
```

# Setup your scripts/Make.defs correctly

- Open Make.defs (still in scripts/ directory) and confirm it will call ld.config for our microcontroller:

```
ifeq ($(CONFIG_ARCH_CHIP_STM32F401RC),y)
LDSCRIPT = ld.script
endif
```

# Rename your board header file

- Now enter inside stm32f401rc-rs485/src/ :
  - $ cd ../src/
  - $ ls

    CMakeLists.txt stm32_adc.c stm32_autoleds.c stm32_buttons.c stm32_spi.c Make.defs stm32_ajoystick.c stm32_boot.c stm32_lcd_ssd1306.c  stm32_userleds.c nucleo-f4x1re.h stm32_appinit.c stm32_bringup.c stm32_mcp2515.c
- Rename nucleo-f4x1re.h to stm32f401rc-rs485.h :
  - $ mv nucleo-f4x1re.h stm32f401rc-rs485.h

# Remove not needed files

- Remove all not needed files for a minimal board support:
  - $ rm stm32_adc.c stm32_ajoystick.c stm32_buttons.c stm32_lcd_ssd1306.c stm32_mcp2515.c stm32_spi.c stm32_userleds.c
- We will end-up with only 7 files:
  - $ ls

    CMakeLists.txt stm32_appinit.c stm32_boot.c stm32f401rc-rs485.h Make.defs stm32_autoleds.c stm32_bringup.c

# Simplify stm32f401rc-rs485/src/CMakeLists.txt

- Open CmakeList.txt and simplify it to:

```
set(SRCS stm32_boot.c stm32_bringup.c)
if(CONFIG_ARCH_LEDS)
  list(APPEND SRCS stm32_autoleds.c)
endif()
if(CONFIG_BOARDCTL)
  list(APPEND SRCS stm32_appinit.c)
endif()
target_sources(board PRIVATE ${SRCS})
if(CONFIG_ARCH_CHIP_STM32F401RC)
  set_property(GLOBAL PROPERTY LD_SCRIPT "${NUTTX_BOARD_DIR}/scripts/ld.script")
endif()
```

# Simplify stm32f401rc-rs485/src/Make.defs

- Open Make.defs and simplify it to:

```
include $(TOPDIR)/Make.defs

CSRCS = stm32_boot.c stm32_bringup.c

ifeq ($(CONFIG_ARCH_LEDS),y)

CSRCS += stm32_autoleds.c

endif

ifeq ($(CONFIG_BOARDCTL),y)

CSRCS += stm32_appinit.c

endif

DEPPATH += --dep-path board

VPATH += :board

CFLAGS += ${INCDIR_PREFIX}$(TOPDIR)$(DELIM)arch$(DELIM)$(CONFIG_ARCH)$(DELIM)src$(DELIM)board$(DELIM)board
```

# Simplify stm32f401rc-rs485/src/stm32_appinit.c

- Open stm32_appinit.c and simplify it to:

```
int board_app_initialize(uintptr_t arg)
{
#ifdef CONFIG_BOARD_LATE_INITIALIZE
  return OK;
#else
  /* Perform board initialization here */

  return stm32_bringup();
#endif
}
```

# Rename the LED name in stm32_autoleds.c

- In stm32_autoleds.c just rename GPIO_LD2 to GPIO_LED1:

```
void board_autoled_initialize(void)
{
  stm32_configgpio(GPIO_LED1);
...
void board_autoled_on(int led)
{
  if (led == 1)
    {
      stm32_gpiowrite(GPIO_LED1, true);
...
void board_autoled_off(int led)
{
  if (led == 1)
    {
      stm32_gpiowrite(GPIO_LED1, false);
```

# Simplify stm32f401rc-rs485/src/stm32_boot.c

- Open stm32_boot.c and simplify the init functions:

```
void stm32_boardinitialize(void)
{
  /* Configure on-board LEDs if LED support has been selected. */
#ifdef CONFIG_ARCH_LEDS
  board_autoled_initialize();
#endif
}
#ifdef CONFIG_BOARD_LATE_INITIALIZE
void board_late_initialize(void)
{
  /* Perform board init here instead of from the board_app_initialize(). */
  stm32_bringup();
}
#endif
```

# Simplify stm32f401rc-rs485/src/stm32_bringup.c

- Open stm32_bringup.c and simplify stm32_bringup() function:

```
int stm32_bringup(void)
{
  int ret = OK;

  return ret;
}
```

# Simplify stm32f401rc-rs485/Kconfig

```
#
# For a description of the syntax of this configuration file,
# see the file kconfig-language.txt in the NuttX tools repository.
#

if ARCH_BOARD_STM32F401RC_RS485

endif # ARCH_BOARD_STM32F401RC_RS485
```

# 1) Add the entry to our board in boards/Kconfig

- Return to the root of nuttx/ and edit boards/Kconfig to add first:

```
config ARCH_BOARD_STM32F401RC_RS485
	bool "STM32F401RC-RS485 Board"
	depends on ARCH_CHIP_STM32F401RC
	select ARCH_HAVE_LEDS
	select ARCH_HAVE_BUTTONS
	select ARCH_HAVE_IRQBUTTONS
	---help---
		This is a minimal configuration that supports low-level test of the
		STM32F401RC-RS485 in the NuttX source tree.
```

# 2) Add the entry to our board in boards/Kconfig

- Add this second part to boards/Kconfig:

```
        default "stm32f401rc-rs485"            if ARCH_BOARD_STM32F401RC_RS485
```

# 3) Add the entry to our board in boards/Kconfig

- Add the third block to boards/Kconfig file:

```
if ARCH_BOARD_STM32F401RC_RS485
source "boards/arm/stm32/stm32f401rc-rs485/Kconfig"
endif
```

# Create a nsh board config

- Rename the directory stm32f401rc-rs485/configs/f401-nsh to nsh
- Edit boards/arm/stm32/stm32f401rc-rs485/configs/nsh/defconfig and replace these CONFIG_ symbols:

```
CONFIG_ARCH_BOARD="stm32f401rc-rs485"
CONFIG_ARCH_BOARD_STM32F401RC_RS485=y
CONFIG_ARCH_CHIP_STM32F401RC=y
```

# Configure and compile your board:

- Configure your board:
  - $ ./tools/configure.sh stm32f401rc-rs485:nsh
- Compile it:
  - make -j

    …

    LD: nuttx

    CP: nuttx.bin
- Check file size:
  - $ ls -l nuttx.bin

    -rwxrwxr-x 1 alan alan 61096 out 15 19:08 nuttx.bin

That’s all folks!